Chapter #

# REPRESENTING ADAPTIVE AND ADAPTABLE UNITS OF LEARNING

*How to model personalized eLearning in IMS Learning Design*

Daniel Burgos, Colin Tattersall and Rob Koper
*Open University of The Netherlands, PO Box 2960, 6401DL, Heerlen*

Abstract: In this chapter we examine how to represent adaptive and adaptable Units of Learning with IMS Learning Design in order to promote automation and interoperability. Based on a literature study, a distinction is drawn between eight types of adaptation that can be classified in three groups: a) the main group, with interfaced-base, learning-flow and content-base; b) interactive problem solving support, adaptive information filtering, adaptive user grouping; and c) adaptive evaluation and changes on-the-fly. Several sources of information are used in adaptation: user, teacher and set of rules. In this paper, we focus on the core group a). Taking the various possible inputs to an eLearning process, we analyze how to model personalized learning scenarios related to these inputs explaining how these can be represented in IMS Learning Design.

Key words: Adaptivity; adaptability; adaptation; personalized learning; IMS Learning Design; Unit of Learning.

## 1. ADAPTIVITY, ADAPTABILITY AND ADAPTATION

### 1.1 What is adaptation?

Recent research has addressed the definitions of adaptivity and adaptability, both focused on personalized learning [1-4]. In summary, adaptivity is the ability to modify e-learning lessons using different parameters and a set of pre-defined rules. In contrast, adaptability is the

possibility for learners to personalize an e-learning lesson by themselves. These two approaches go from machine-centered (adaptivity) to user-centered (adaptability). However, we contend that there are a vast number of stages in between which define a gray area, with some adaptivity and some adaptability. In practice, it is quite difficult to isolate one from the other due to their close relationship [5]. Furthermore, all the in-between stages are also personalized learning as they enable the dynamic adaptation of several features in a course, a lesson plan or a Unit of Learning. As a result, we view the concepts not as two opposite corners from which to look at personalized learning, but as describing a wide range of approaches taking the best of each. Hereafter, we use the word *adaptation* to cover the various approaches.

## 1.2 Adaptation by whom?

The Adaptive Hypermedia Systems approach indicates that the best adaptation is the one that a user cannot see; the one where the cognitive load of the user is reduced to the minimum or even zero [5,6]. For example, a user is able to follow an introductory section in a course only the first time he logs-in, since that section is hidden from the second login onwards. This means that the adaptation must be done by the system automatically, depending on the user's performance. From this point of view, mainly based on adaptive hypermedia Web systems, there are two well-defined inputs: user and set of rules This approach is focused on *micro-adaptation*: monitoring the learning behavior of the student while running specific tasks and adapting the instructional design afterwards, based on quantitative information [7,8]. On the one hand, there is a provider of some information based on behavior (the student). This means a non-voluntary action where the information is collected from the provider without his knowledge or agreement. On the other hand, there is a decision maker (the set of rules) who takes the collected information and decides further movements based on it.

Although this is an interesting approach, there are some drawbacks on this view of the involved inputs. First, the input of the student as a decision maker, who progresses from one side, being a non-voluntary provider of information, to the opposite side, stressing the cognitive load in the final decisions about the learning process. We agree that sometimes a student cannot decide what is the best or the most convenient for him at a certain moment, because of lack of training or of a broader vision on his studies [9]. But we also hold that, leaving apart his interests and opinions, this approach dismisses his motivations, wishes and personal drives (i.e. feelings and emotions). All of them are important parts in the learning process. Second, the input of the teacher as one of the participants involved while building

adaptive courses is something to take into consideration. His inputs, feedbacks and contributions do not end when the course is delivered. A teacher can and should modify the learning flow of a student or a group of students as long as the course takes place. He should make a good balance between the time and the effort invested and the actual outcome, though.

Furthermore, we count up to four inputs in a balanced formula for adaptation: a) the user, when some information is taken from his behavior and performance [10,11]; b) the user, when he contributes with his/her own personal decision [12]; c) the teacher, when he also contributes with his/her personal decision [13]; and d) the set of pre-defined rules made by the learning designer/author (usually, the same teacher); a kind of adaptation engine [14]. The inputs a) and b) provide some personal needs and drives to learn; the inputs c) and d) take care of the didactical quality and of the learning efficiency.

## 2. ELEMENTS IN IMS LEARNING DESIGN TO REPRESENT ADAPTATION

IMS Learning Design [15], or simply IMS LD, is a specification focused on the modeling of Units of Learning (UoLs). IMS LD augments other well-known e-learning specifications, like SCORM [16], IMS Content Packaging [17], IMS Question and Test Interoperability [18] or IMS Simple Sequencing [19]. IMS LD has also a good understanding with some e-learning systems like Moodle [20]. IMS LD provides a language to describe the teaching and learning process in a Unit of Learning. It describes among other things the roles, the activities, the basic information structure, the communication among different roles and users; and all these under the pedagogical approach decided by the teacher and/or the learning designer.

IMS LD is technically structured in three levels: Level A provides the basic definition of the Unit of Learning, meaning learning activities, support activities, environments, roles, role-parts, learning flow and activity structures. Level B adds some important elements to Level A: properties, conditions, calculations, monitoring services and global elements, and Level C adds the use of notifications.

Level A provides a basic and larger information structure and the guidelines to assign activities to roles and to define learning acts and plays. Level B provides a more flexible use of these elements since it allows for the modeling of alternative learning itineraries, dynamic feedback, run-time tracking and collaborative learning, amongst others [21]. Level B is the key to the most expressive adaptation in IMS Learning Design as it provides some powerful resources and elements to model personalized Units of

Learning. The elements in Level B that can be used for adaptation are explained below: Properties, global elements, monitoring service, conditions and calculations.

## 2.1 Properties

Properties in IMS LD are variables that are declared in the learning design (in the *imsmanifest.xml* file) and are used and viewed using global elements written in XHTML files. They can also be used inside the learning design, in conditions or calculations. We deal with the global elements, the conditions and the calculations in the next sections. The values inside of a property can be of different types like Integer, Boolean, text, float, file, etc.

There are five types of properties in IMS LD:

1. *Local properties*. Only accessible in the instance of a Unit of Learning (the *run*) [22]. They have the same value for every user in that instance. For example, a question in a questionnaire.

2. *Local personal properties*. Only accessible in the instance of a Unit of Learning but they have a different value for every user. For instance, the name of a student.

3. *Local role property*. Only accessible in the instance of a Unit of Learning and it has a different value for every role. For instance, two different roles, meaning two different groups, with a different approach to a problem.

4. *Global personal properties*. Defined in the Unit of Learning, and set-up and viewed in every run of this Unit of Learning for the owner of the property. Each user has different values. For instance, personal information of a student in a school.

5. *Global properties*. A constant value, set-up in one instance of a Unit of Learning and viewed in every instance by all the people who have access to it. For instance, a numeric threshold in an evaluation.

## 2.2 Global elements

Global elements are included in external XML or XHTML files to read or to write the property values that are defined in a learning design. These files must be of type *imsldcontent* to prepare for appropriate processing in the IMS LD engine. Global elements are used basically to address two things: information layers and properties. If we use information DIV layers, some information can be prepared and conditionally shown or hidden. If we use properties, once a property is correctly declared in the manifest file, it can be shown (*view-property*) and it can also be given a value (*set-property*). For instance, the request of a name, or even the request of several fields in a

form, collected in a group of properties. In order to set a property, a specific user interface control or form is generated depending on the type of property (integer, text, file, etc.).

### 2.3 Monitoring service

A monitoring service provides the possibility to track the progress of users or groups of users and of the properties related to them. This service can be set-up to define which role can access the information. For instance, the teacher role monitors a student role, allowing for a tracking of the different properties related to the student role, such as grades, remarks and completion of activities.

### 2.4 Conditions

Conditions in IMS LD have an *if-then-else* structure. We can use this structure to chain a series of conditions and create a more complex rule. Conditions in IMS LD are largely used to adapt learning flows, contents and other aspects of personalized learning, as we will explain later on. The rules are defined at design-time (when the UoL is modeled) and they are evaluated at run-time (when the UoL is executed), leading to content which is adapted dynamically.

### 2.5 Calculations

Calculations are basic arithmetic in IMS LD. Inside the learning design section of a manifest we can sum, subtract, multiply and divide numbers and values of properties to store the final result in another property. Later, this property can be seen using the global element *view-property* in a XML external file, already described.

## 3. MAIN TYPES OF ADAPTATION AND RELATED REPRESENTATION WITH IMS LD

The literature describes eight types of adaptation, divided into three groups [4]. The first group a) points to learning flow, content and interface [23-26,1,27]; the second group b) is based on interactive problem solving support, adaptive information filtering and adaptive user grouping [23,26]; the third group c) consists of adaptive evaluation and changes on-the-fly [28,29]. The groups b) and c) could be considered as subgroups of a), as they make use of the types in a) to define and feed themselves. Hereafter, we will

focus our research on the first group of adaptation (learning flow, content and interface based), explaining what these approaches consist of and how to represent them in IMS LD.

## 3.1 Adaptation based on the learning flow

The modification of the learning flow as the Unit of Learning is being executed is one of the most often used types of adaptation. Taking the flow as a base, the Unit of Learning provides different activities, resources and services, depending on these four inputs during execution (user´s behavior and performance, user´s decision, teacher and set of rules). The activity structure in an IMS LD UoL is defined using plays, acts, activity structures, learning activities, support activities and environments. We can also use the property of *visibility* to hide and show these elements and to adapt the learning flow. In these cases the property works as a flag, switching on and off the elements referred to We now show five scenarios and their related implementations of learning-flow based adaptation focused on the several possible inputs. The pseudo XML code shown is an abstraction of the IMS LD original source, concentrating on the key elements of the specification needed for a more self-understandable explanation; all the examples can be found at [30].

### 3.1.1 The set of rules modifies the learning flow taking the user's behavior as an input

Scenario: One activity is shown the very first time that a user logs into the Unit of Learning, and it remains hidden from the second time onwards. User and the set of rules are the inputs involved.

Implementation: We create a personal local property (*Prop-Firsttime*) (type Boolean) and it is initialized to 0. When the user logs-in the first time the property is set to 1. A condition shows a specific activity (*LA-FirstActivity*) only when this flag property is 0:

```
<locpers-property identifier="Prop-Firsttime ">
  <title>PropFirsttime</title>
  <datatype datatype="boolean"/>
  <initial-value>0</initial-value>
</locpers-property>
```

```
<if>
  <is>
    <property-ref ref="Prop-Firsttime"/>
    <property-value>0</property-value>
  </is>
</if>
<then>
  <show>
```

```
      <activity-structure-ref ref=LA-FirstActivity"/>
    </show>
    <change-property-value>
      <property-ref ref="Prop-Firsttime "/>
      <property-value>1</property-value>
    </change-property-value>
  </then>
  <else>
    <hide>
      <activity-structure-ref ref=LA-FirstActivity"/>
    </hide>
  </else>
```

### 3.1.2 The set of rules adapts the learning flow based on the user's performance

Scenario: The behavior of a user is a possible input. Also, the performance and the cognitive load of the user during an activity could adapt the learning flow. This example, *GeoQuiz3*, provides a general quiz on Geography with five questions and multiple answers. The user gets a score, an average and an accuracy measure. The subsequent activity to be studied by the student depends on these results, and it is taken from four possible activities, including the repetition of the task if a certain threshold is not reached. Therefore, user and engine are the inputs involved.

Implementation: This example guides the learning flow of the student based on his performance and on a set of pre-defined rules. Also, the pre-defined algorithm interlaces properties, activities and conditions to get the final result. First, a set of properties is arranged:

```
  <locpers-property identifier="Value1">
    <datatype datatype="integer"/>
    <initial-value>0</initial-value>
  </locpers-property>
  <locpers-property identifier="Question1">
    <datatype datatype="string"/>
    <initial-value>Select</initial-value>
    <restriction type="enumeration">Select</restriction>
    <restriction type="enumeration">Malasia</restriction>
    <restriction type="enumeration">The Moon</restriction>
    <restriction type="enumeration">Canada</restriction>
  </locpers-property>
```

Second, every question is included in an external XML file, using the global element *set-property*. In this case, HTML code and IMS LD code are combined:

```
  <html>
    <td>Where is the Mare Tranquilitatis?</td>
    <td><p>
      <set-property ref="Question1" property-of="self" view="value"/>
    </p></td>
  </html>
```

And third, the conditions are established in the learning design to check the results and to define the adaptive feedback and the next activity to be undertaken. For instance, by changing the completion value of a question, as follows:

```
<if>
   <is>
      <property-ref ref="Question1"/>
      <property-value>The Moon</property-value>
   </is>
</if>
<then>
   <change-property-value>
      <property-ref ref="Value1"/>
      <property-value>1</property-value>
   </change-property-value>
</then>
```

### 3.1.3 The user himself modifies the learning flow based on his personal decision

Scenario: In *Learning to Listen to Jazz* the user can choose the learning itinerary out of two possible paths: Historic and thematic. The user can swap between both at three different points in the learning flow. The activities already done in one path remain in the same state when the user moves to the alternative path. Therefore, they are the same activities but with two different ways of study. In this case, the adaptation comes from the user, based on a pre-design of the course by the author/tutor.

Implementation: Both paths are predefined and are shown and hidden depending on the value of the Boolean property *SelectionOfRoute* that the user can change on request:

```
<if>
   <no-value>
      <property-ref ref="SelectionOfRoute"/>
   </no-value>
</if>
<then>
   <hide>
      <activity-structure-ref ref="AS-Thematic"/>
      <activity-structure-ref ref="AS-Historic"/>
   </hide>
</then>
<else>
<if>
  <is>
    <property-ref ref="SelectionOfRoute"/>
    <property-value>Thematic</property-value>
  </is>
</if>
<then>
   <show>
      <activity-structure-ref ref="AS-Thematic"/>
   </show>
   <hide>
      <activity-structure-ref ref="AS-Historic"/>
```

```
    </hide>
  </then>
```

### 3.1.4 The teacher modifies the learning flow of the user

Scenario: The teacher monitors the performance of a user or of a group of users and he decides which activities should be shown and hidden and in which order. The adaptation comes from the teacher, taking into consideration several other inputs from the user or group of users.

Implementation: Two things are needed. The first one is a monitoring service *S-Performance* defined in an environment in the learnign design that allows for the observation of every user. The second one is the definition of a set of flag properties (personal or role properties). These properties show and hide activities, structures and environments to the end-user/s, e.g. *FlagForActivity1*. The definition is made in the *imsmanifest.xml* file; the actual view and set-up is made in an external XML file using global elements (*view-property* and *set-property*):

```
<environment identifier="E-Performance">
  <title>You can watch the performance of every user</title>
  <service identifier="S-Performance">
    <monitor>
      <role-ref ref="Tutor"/>
      <title>Tracking personal performance</title>
      <item identifierref="R-Performance"/>
    </monitor>
  </service>
</environment>
```

```
<set-property ref="FlagForActivity1" property-of="supported"/>

<view-property ref="StudentPerformance" property-of="supported"/>
```

### 3.1.5 Integrated approach

A last scenario is based on the integrated decision of several of the previous scenarios agreed by consensus. More than one input (user, teacher, set of rules) are taken at the same time and the final decision leans on one of them. For example, a teacher can take the suggestion of a learner, his behavior, his performance, the recommendation of a set of rules and make the final decision. Alternatively, the engine takes the role of the teacher when making the final decision. Or even the student takes the suggestions of the teacher and of the engine and decides what to carry out next.

## 3.2 Adaptation based on the content

In the previous section, we saw that a learning flow is mainly focused on the sequence of the activities in a Unit of Learning. However, content based adaptation is focused on the information of every activity, and on the activity itself. There are two main approaches for content based adaptation: Flag properties and content of properties.

### 3.2.1 Flag properties

This approach is focused on the use of flag properties that switch on and off a certain information layer (such as a DIV layer in XHTML).

Scenario: A student follows a questionnaire. The right answer to every sequenced question is the key to read the next question (example *GeoQuiz1*). Depending on the answer, some support information is shown.

Implementation: The actual definition of the layers (*Answer1_Wrong* and *Answer1_Right*) and the information inside them is defined in external XML file/s. These files are linked to the *imsmanifest* and identified as resources of type *imsldcontent*; they also have the layers to be shown and hidden and they use global elements:

```
<html>
     <h1>Question 1/5</h1>
     <p>Where is the Eiffel Tower?
     <blockquote><b>A</b> Paris</blockquote>
     <blockquote><b>B</b> Brussels</blockquote>
     </p>
     <p>Your answer is:
          <set-property ref="Answer1" property-of="self"/></p>
     <div class="Answer1_Wrong">
        <p>Choose another answer.
          '<view-property ref="Answer1" />' is not right</p>
     </div>
     <div class="Answer1_Right">
        <p>Congratulations! It's the right answer</p>
        <img src="eiffel.jpg"></img>
     </div>
</html>
```

The definition of the method is in the *imsmanifest.xml* file. The definition and initialization of the flag properties and the learning activities are also done in this file, as well as the management of the visibility of the DIV layers described in the external files. When *QuestionTrue1* turns to 1 the first activity *question1* is finished:

```
<locpers-property identifier="QuestionTrue1">
  <datatype datatype="boolean"/>
  <initial-value>0</initial-value>
</locpers-property>
```

```
<learning-activity identifier="question1">
    <title>Question 1</title>
    <activity-description>
       <item identifierref="res-question1"/>
    </activity-description>
    <complete-activity>
       <when-property-value-is-set>
          <property-ref ref="QuestionTrue1"/>
          <property-value>1</property-value>
       </when-property-value-is-set>
    </complete-activity>
</learning-activity>
```

```
<if>
  <is>
     <property-ref ref="Answer1"/>
     <property-value>A</property-value>
  </is>
</if>
<then>
  <hide>
     <class class="Answer1_Wrong" />
  </hide>
  <show>
     <class class="Answer1_Right"/>
  </show>
  <change-property-value>
     <property-ref ref="QuestionTrue1"/>
     <property-value>1</property-value>
  </change-property-value>
</then>
```

### 3.2.2 Content of properties

The second approach to adaptation allows for the modification of the content of a property that has been pre-defined inside an activity. In this case, the property of visibility remains always on, but the content of the field changes. We need two steps, therefore. One is at design-time - making the definition and configuration of the property, done in the learning design of the *imsmanifest* file. The second step, at run-time, involves changing the content of the property, made in an external XML file. We use the global elements *set-property* and *view-property* to configure and see the content of the field/property. In the example *GeoQuiz3* used above, the property with the adaptive feedback (*prop-feedback*) is shown after the completion of the form:

```
<td><p>Your adaptive feedback is: <view-property ref="prop-
feedback" property-of="self" view="value"/></p></td>
```

Depending on the final score, the content of the property (*prop-feedback*) is different, though:

```
<if>
  <and>
     <greater-than>
```

```
          <property-ref ref="score"/>
          <property-value>49</property-value>
        </greater-than>
        <less-than>
          <property-ref ref="score"/>
          <property-value>76</property-value>
        </less-than>
      </and>
    </if>
    <then>
      <change-property-value>
        <property-ref ref="prop-feedback"/>
        <property-value>Well done! Your score is promising. You are in
Level 2</property-value>
      </change-property-value>
    </then>
```

Another possible scenario is set-up when a teacher changes the content of some fields dynamically while executing the Unit of Learning. In *Quo Builder2* the student can see the questions and possible answers in a questionnaire as long as the teacher defines them. The teacher can also design the basic configuration of the form: the general welcome messages, the adaptive feedback and the scoring system. At the end, it becomes an interactive and dynamic evaluation test, modified at run-time.

## 3.3 Adaptation based on the interface

Interface based adaptation is quite different to content based adaptation. Content adaptation is based on the information inside an activity that is shown and handled. Interface adaptation is based on options, navigation and visualization facilities. In [4] the authors state that interface adaptation is not possible with today's tools for IMS LD, such as CopperCore Player [31], Reload LD Player [32] and Sled [33], or the editors CopperAuthor [34] and Reload LD Editor [32]. As long as the adaptation of the interface is based on the tool and not on the Unit of Learning that is interpreted by the player, this is still true. Today's players do not yet provide facilities to change the size or the position of the navigation panels, or even open and close the working areas in the player. Either, these tools cannot change the style sheets related to a HTML file, part of the content, and any of the linked features, as font-size, font-type or background color, for instance. Although the CopperCore engine provides the appropriate infrastructure, no player uses it so far.

Nevertheless, some kind of adaptive interface is possible, using DIV layers and environments.

Scenario: The options and the look and feel of an interface are adapted on the user's request.

Implementation: Regarding activities, several DIV layers or learning activities can be set-up with a different visualization for the same content.

For instance, linking the same file to different CSS style sheets. First, we define the different activities:

```
<learning-activity identifier="Activity1InterfaceA">
   <title>Question 1</title>
   <activity-description>
      <item identifierref="firstlessonInterfaceA"/>
   </activity-description>
</learning-activity>

<learning-activity identifier="Activity1InterfaceB">
   <title>Question 1</title>
   <activity-description>
      <item identifierref="firstlessonInterfaceB"/>
   </activity-description>
</learning-activity>
```

Later, we link the CSS style sheets with the same file lesson1.html, resulting in two different resource identifiers:

```
<resource    identifier="firstlessonInterfaceA"    type="webcontent"
href="lesson1.html">
   <file href="lesson1.html" />
   <file href="stylesheetA.css" />
</resource>

<resource    identifier="firstlessonInterfaceB"    type="webcontent"
href="lesson1.html">
   <file href="lesson1.html" />
   <file href="stylesheetB.css" />
</resource>
```

And finally, we show and hide the activity linked to the related resource:

```
<if>
   <is>
      <property-ref ref="InterfaceToChoose"/>
      <property-value>A</property-value>
   </is>
</if>
<then>
   <show>
      <activity-structure-ref ref="firstlessonInterfaceA"/>
   </show>
   <hide>
      <activity-structure-ref ref="firstlessonInterfaceB"/>
   </hide>
</then>
```

Another possibility could be to adapt not only the look and feel of a DIV layer, but also its content and the options of interaction inside it, resulting in a block of information and interaction.

Last one additional set-up is to define different environments with several contents and services and link them to different activities or activity structures. They will be shown/hidden together with the related activity. This approach can be managed following any of the methods aforementioned.

Furthermore, we could count different services, contents and options in every environment that are depending on the learning tree. To some extent, this means a *de facto* sub-division on the screen and a different adapted interface. So, one activity linked to one environment, and both are shown or hidden, resulting in a final personalized interface:

```
<show>
   <environment-ref ref="ENVfirstlessonInterfaceA"/>
</show>
<hide>
   <environment-ref ref="ENVfirstlessonInterfaceB"/>
</hide>
```

## 4. CONCLUSION

In this chapter we have shown three main types of adaptation with some typical scenarios and their related implementations, based on the learning flow, the content and the interface. Also, we have stated the four different inputs involved in the adaptation process: a) the user, based on his behavior and his performance; b) the user, based on his personal decision; c) the personal decision of the teacher; and d) the set of rules in an engine, pre-defined by a learning designer. To implement these scenarios in IMS LD we use the basic structure that Level A provides and the core elements of Level B: properties, global elements, monitoring service, conditions and calculations.

We conclude that it is possible to represent strategies for adaptation taken from all these inputs and types, in IMS LD. Whether we talk about adaptivity or about adaptability, the issue of personalized learning can be modeled with the specification, using different approaches in order to support learners build better competences and skills.

## ACKNOWLEDGEMENTS

This chapter is partially supported by the European Projects ProLearn [35] and TENCompetence [36].

## REFERENCES

1. A. Ahmad, O. Basir, and K. Hassanein, Adaptive user interfaces for intelligent e-Learning: issues and trends, in: *The Fourth International Conference on Electronic Business, ICEB2004*, (Beijing, 2004).

2. S.Y. Chen, and G.D. Magoulas, *Adaptable and Adaptive Hypermedia Systems* (IRM Press, Hershey, PA, 2005).
3. A. Cristea, Authoring of Adaptive Hypermedia, *Educational Technology & Society*, 8, 6-8 (2005).
4. D. Burgos, C. Tattersall, and R. Koper, Representing adaptive eLearning strategies in IMS Learning Design, in: *TENCompetence*, (Sofia, Bulgaria, 2006).
5. M. Klann, The EUD-Net's Roadmap to End-User Development, in: *Workshop on End User Development at ACM CHI 2003 Conference*, (Fort Lauderdale, USA, 2003).
6. P. De Bra, N. Stash, and D. Smits, Creating Adaptive Web-Based Applications, in: *10th International Conference on User Modeling*, (Edinburgh, Scotland, 2005).
7. O. Park, and J. Lee, Adaptive Instructional Systems, *Educational Technology Research and Development*, 25, 651-684 (2003).
8. F. Mödritscher, V. García, and C. Gütl, The Past, the Present and the Future of adaptive E-Learning. An Approach within the Scope of the Research Project AdeLE, in: *ICL*, (Villach, Austria, 2004).
9. R. Snow, Aptitude, learner control and adaptive instruction, *Educational Pshychologist*, 15, 151-158 (1980).
10. R.D. Tennyson, Instructional control strategies and content structure as design vairables in concept acquisition using computer-based instruction., *Journal of Educational Psychology*, 72, 525-532 (1980).
11. R.D. Tennyson, Use of adaptive information for advisement in learning concepts and rules using computer assisted instruction, *American Educational Research Journal*, 18, 425-438 (1981).
12. B. Bell, and S. Kozlowski, Adaptive guidance: Enhancing self-regulation, knowledge, and performance in technology-based training, *Personnel Psychology*, 55, 267-306 (2002).
13. P. Van Rosmalen, H. Vogten, R. Van Es, P. Van, H.,, P. Poelmans, and R. Koper, Authoring a full life cycle model in standards-based, adaptive e-learning, *Educational Technology & Society*, 9, 72-83 (2006).
14. B. Wasson, Advanced educational technologies: The learning environment, *Computers in Human Behavior*, 13, 571-594 (1997).
15. IMSLD, IMS Learning Design (February 27th); http://www.imsglobal.org/learningdesign/index.cfm.
16. ADL, Sharable Object Reference Model, SCORM (May 9th, 2006); http://www.adlnet.org/index.cfm?fuseaction=Scormabt.
17. IMSCP, IMS Content Packaging (May 10th, 2006); www.imsglobal.org.
18. IMSQTI, IMS Question and Test Interoperability (May 10th, 2006); www.imsglobal.org.
19. IMSSS, IMS Simple Sequencing [www.imsglobal.org] (May 10th, 2006); www.imsglobal.org.
20. M. Dougiamas, Moodle (May 10th, 2006); http://moodle.org.

21. R. Koper, and D. Burgos, Developing advanced units of learning using IMS Learning Design level B, *International Journal on Advanced Technology for Learning*, 2, (2005).
22. C. Tattersall, H. Vogten, F. Brouns, R. Koper, P. van Rosmalen, P. Sloep, and J. van Bruggen, How to create flexible runtime delivery of distance learning courses, *Educational Technology & Society*, 8, 226-236 (2005).
23. R. Baeza-Yates, and B. Ribeiro-Nieto, *Modern Information Retrieval* (Addison-Wesley, Boston, MA, USA, 1999).
24. P. Brusilovsky, and P. Miller, Course Delivery Systems for the Virtual University, in: *Access to Knowledge: New Information Technologies and the Emergence of the Virtual University*, Tschang F.T., and T. Della Senta (Eds.), (Elsevier Science and International Association of Universities, Amsterdam, 2001), pp.167-206.
25. D. Chin, Empirical Evaluation of User Models and User-Adapted Systems., *User Modeling and User-Adapted Interaction*, 11, 181-194 (2001).
26. P. Brusilovsky, and C. Paylo, Adaptive and Intelligent Web-based Educational Systems, *International Journal of Artificial Intelligence in Education*, 13, 156-169 (2003).
27. P. De Bra, L. Aroyo, and A. Cristea, Adaptive Web-based Educational Hypermedia, in: *Web Dynamics, Adaptive to Change in Content, Size, Topology and Use*, M. Levene, and A. Poulovassilis (Eds.), (Springer, 2004), pp.387-410.
28. A. Merceron, and K. Yacef, A Web-based tutoring tool with mining facilities to improve learning and teaching, in: *AI-Ed'2003*, U. Hoppe, F. Verdejo, and J. Kay (Eds.) Vol. (IOS Press, 2003).
29. P. Van Rosmalen, and J. Boticario, Using Learning Design to support design- and runtime adaptation, in: *Learning Design: A Handbook on Modeling and Delivering Networked Education and Training*, R. Koper, and C. Tattersall (Eds.), (Springer Verlag, Heidelberg, Germany, 2005).
30. LN4LD, Units of Learning developed by several authors at Learning Network for Learning Design of The Open University of The Netherlands (March 6th); http://moodle.learningnetworks.org/course/view.php?id=20 and http://dspace.ou.nl.
31. H. Vogten, and H. Martens, CopperCore 3.0 (April 13th, 2006); www.coppercore.org.
32. Bolton, Reload Project (April 16th, 2006); www.reload.ac.uk.
33. OUUK, Sled player (April 13th, 2006); http://sled.open.ac.uk.
34. W. Van der Vegt, CopperAuthor (April 13th, 2006); www.copperauthor.org.
35. Pro-Learn, Pro-Learn Project (July 31st, 2006);
36. TENCompetence, TENCompetence Project (July 31st, 2006); www.tencompetence.org.